Randomized Skip Lists-Based Private Authentication for Large-Scale RFID Systems

Kazuya Sakai[1] Min-Te Sun[2] Wei-Shinn Ku[3] Ten H. Lai[1]
[1]Department of Computer Science and Engineering, The Ohio State University, OH 43210, USA
[2]Department of Computer Science and Information Engineering, National Central University, Taiwan
[3]Department of Computer Science and Software Engineering, Auburn University, AL 36849, USA
{sakai,lai}@cse.ohio-state.edu, msun@csie.ncu.edu.tw, weishinn@auburn.edu

ABSTRACT

The performance of key authentication and the degree of privacy in large-scale RFID systems are considered by many researchers as tradeoffs. Based on how keys are managed in the system, the privacy preserving tag authentications proposed in the past can be categorized into tree-based and group-based approaches. While a tree-based approach achieves high performance in key authentication, it suffers from the issue of low privacy should a fraction of tags be compromised. On the contrary, while group-based key authentication is relatively invulnerable to compromise attacks, it is not scalable to the large number of tags. In this paper, we propose a new private tag authentication protocol based on skip lists, named Randomized Skip Lists-based Authentication. Without sacrificing the authentication performance, our scheme provides a strong privacy preserving mechanism.

Categories and Subject Descriptors

H.2.7 [**Database Administration**]: Security, integrity, and protection

General Terms

Security

Keywords

RFID, privacy, authentication

1. INTRODUCTION

Radio Frequency Identification (RFID) is widely used to smooth the way of various applications, such as library managements [7], transportation payment, natural habitat monitoring, indoor localization [5, 11, 12], and so on. In these systems, the administrator manages and monitors a large number of objects by reading passive RF tags attached to the objects with an RF reader. To protect the tag's content [9], low-cost cryptographic operations [4] are conducted during singulation process. Hence, on receiving the tag's reply, the reader must scan all keys to find the corresponding key in order to decrypt the content. When it comes to a large-scale RFID system, the authentication process can take a long time.

To accommodate this issue, a number of private tag authentication protocols with structured key management have been proposed. In these approaches, a unique key and a set of group keys are assigned to each tag. The group keys are shared among several tags and are used to confine the search space of the unique key corresponding to a tag's reply. Based on how group keys are managed, they are categorized into two types: tree-based [2, 6, 7] and group-based protocols [1, 3]. In a tree-based protocol, tags are mapped to leaf nodes in the tree and keys are assigned to internal nodes. Each tag has its unique key and a set of shared keys associated with the nodes from the leaf to the root. By traveling the tree, the reader can securely singulate tags. This results in high authentication efficiency, but discloses a large amount of information once tags in the system are compromised. On the contrary, in a group-based protocol, each tag has two kinds of keys: a unique key and a group key. With this approach, even if one of the group members is compromised, tags in other groups are intact. However, the authentication efficiency of this approach is low.

Therefore, for large-scale RFID systems, the performance and privacy/security of key authentication are commonly seen as tradeoffs. In this research, we propose a scheme that provides both good performance and a high level of privacy/security for a large-scale RFID system. Since both tree-based and group-based structures have pros and cons, we take a different approach based on *skip lists* [8], a data structure with which operations are performed in a logarithmic order like a balanced tree. We propose a new private tag authentication protocol, named Randomized Skip Lists-based Authentication (RSLA), which provides strong privacy protection and high performance of authentication like the tree-based approach. In our proposed scheme, an interrogator authenticates a tag by traveling skip lists from top to bottom with a random rotation at each level.

The rest of this paper is organized as follows. In Section 2, we propose RSLA. Section 3 concludes this paper.

MobiHoc'13, July 29–August 1, 2013, Bangalore, India.

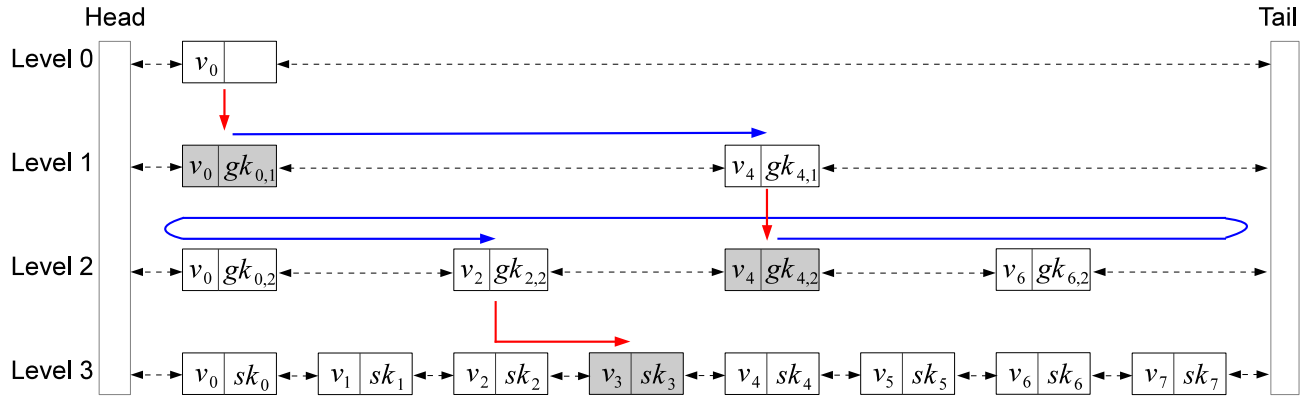

Figure 1: An example of authentication.

2. PRIVATE AUTHENTICATION PROTOCOL

2.1 Protocol Overview

In this paper, we propose Randomized Skip Lists-based Authentication (RSLA) which consists of four components: key issuing (initialization), private authentication, key-updating, and system maintenance.

In the key issuing process, the system generates skip lists. RF tags are randomly assigned to nodes in the lowest level list. A unique key and a set of group keys are assigned to each tag by traveling from a node at the bottom to the top level list. In the authentication, an RF reader scans group keys to narrow the search space of the corresponding unique key for a tag by traveling from the top list to the bottom list.

Due to the space constraint, we omit the discussion of key-updating and system maintenance in this paper. Interested readers please refer to our technical report [10] for detail.

2.2 Def nitions and Assumptions

In our assumptions, an RFID system consists of N tags and a reader, which is connected to the back-end server. For simplicity, it is assumed that the reader and the back-end server can securely communicate, and thus the reader is the final destination of a tag's data.

n_r and n_t represent nonce randomly selected by the reader and a tag, respectively. For a given key K and an input x, the hash function $H(x)$ is assumed to be collision resistant, and an encryption function $E(K, x)$ is implemented by low-cost cryptographic operations [4]. A reader is assumed to have enough computational power to run a decryption function $D(K, x)$ with a key K and an input x.

2.3 Construction of Skip Lists

To construct skip lists for key management, we modify the construction process as follows. Instead of randomly selecting nodes that appear at the list in the upper levels, we deterministically select nodes to keep the number of nodes at each level consistent.

Let L_i be the list at the i-th top level. Each list consists of a set of nodes. A node i, denoted as v_i, has pointers to left and right nodes in the same list, which are denoted by $v_i.left$ and $v_i.right$. The left pointer of the first node and the right pointer of the last node are null. In addition, the pointers to the first and last nodes of list L_i are kept in $L_i.head$ and $L_i.tail$.

We generate skip lists that contain $\eta + 1$ lists. Each list L_i contains k^i nodes, where η is defined as $\lceil log_k(N) \rceil$ so that we can map all tags to the nodes in the lowest level list. Note that if there are more than N nodes, some nodes are not assigned a tag. Given the number of tags N and a balancing factor k, a list L_η with k^η nodes is first created. Then, node v_i is added into $L_{\eta-1}$ if $i \ mod \ k = 0$. For each level j, node v_i $(0 \leq j \leq \eta - 1)$ is added into L_j if $i \ mod \ k^{\eta-j} = 0$. This process is repeated from η to 0. The top level list always has one node, i.e., $L_0 = \{v_0\}$, since the number of nodes at the lowest level list is k^η.

Each node in skip lists has a set of keys. We define $v_i.key[j]$ as the variable to store node v_i's key for Level j. If v_i does not appear in L_j, $v_i.key[j]$ is empty. Assuming Tag t is mapped to v_i, the unique key sk_t of Tag t is located at $v_i.key[\eta]$. Let us denote $gk_{i,j}$ the group key, which is stored at $v_i.key[j]$. Thus, all nodes in skip lists have a unique key in $v.key[\eta]$, and group keys for Level j $(1 \leq j \leq \eta - 1)$ in $v.key[j]$ if v appears in L_j. We do not assign any key to the node in the top level list L_0, since L_0 has only one node. Thus, $v_0.key[0]$ is empty.

Since the construction of skip lists is deterministic, our skip lists with factor k work in similar fashion as a k-balanced tree. The reason why we employ skip lists instead of a balanced tree is that the link among the nodes in the same level is utilized for random rotation. Thus, we do not have

Table 1: Definition of notations.

Symbols	Definition
k	The balancing factor of skip lists
N	The number of tags in the system
η	The height of skip lists, $\lceil log_k N \rceil$
L_i	The list at Level i in skip lists $(0 \leq i \leq \eta)$
v_i	Node i in a list
sk_i	Tag i's unique secret key
GK_i	A set of group keys of Tag i, $\{gk_1, gk_2, ..., gk_{\eta-1}\}$
R_i	A set of random numbers of Tag i, $\{r_1, r_2, ..., r_{\eta-1}\}$
n_t, n_r	Nonces from a tag and a reader
β	Tag's reply, $\{\beta_1, \beta_2, .., \beta_\eta\}$
N_c	The number of compromised tags in the system
N_g	The number of compromised tags in a group
$E(.)$, $D(.)$	The encryption and decryption functions
$H(.)$	The hash function
A	System anonymity
S_i	Anonymous set that Tag i belongs to

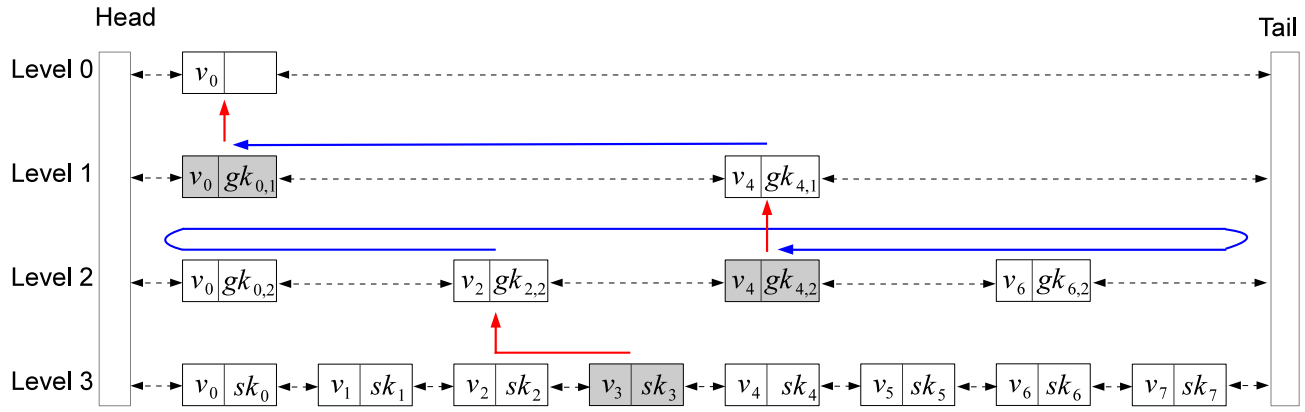

Figure 2: An example of key issuing.

to modify the data structure of skip lists to achieve the design goals.

2.4 Key Issuing

In RSLA, Tag t has three variables, the unique secret key sk_t, a set of group keys GK_t, and a set of random numbers R_t. Each tag t is randomly assigned to a node, say v_i, in the lowest level list L_η. Starting from v_i, the key issuer traverses to L_0 by shifting to the left for r_j nodes at each L_j $(1 \leq j \leq \eta - 1)$, where r_j is randomly chosen between 0 and $|L_j| - 1$ (i.e., $k^j - 1$). By doing this, the key of the selected node for each level is assigned to a tag.

At v_i in L_η, Tag t obtains the unique key from $v_i.key[\eta]$, and then the pointer moves to $L_{\eta-1}$. When v_i in L_η does not appear in $L_{\eta-1}$, the pointer first moves to node v_j where $j = i - i \ mod \ k$, and then moves to $L_{\eta-1}$. In general, for the current node v_i in L_m, the pointer moves to v_j where $j = i - i \ mod \ k^{\eta-m+1}$, and then goes to L_{m-1}. Thus, this can be seen as a parent and children relation of a k-balanced tree, i.e., v_j in L_{m-1} has k children v_i in L_m $(j \leq i \leq j + k - 1)$.

Every time, the pointer arrives at a upper level list, for instance L_j, the key issuer takes the left shift by r_j at L_j $(1 \leq j \leq \eta - 1)$. Here, r_j is randomly selected between 0 and $|L_j| - 1$, and added to set R_t. Note that the left shift is not taken at the L_η and L_0. The shifting can be done by moving the pointer via $v_i.left$. If $v_i.left = null$, i.e., v_i is the first node in L_j, the pointer moves to $L_j.tail$, i.e., the last node in L_j. Let v_i be the node in L_j after shifting. Tag t obtains the group key from $v_i.key[j]$. Then, the pointer moves to the upper level. This process continues until the key issuer reaches L_0.

At the end of this process, Tag t has one unique key, $\eta - 1$ group keys, and $\eta - 1$ random numbers. The pseudo code of the key issuing is given in Algorithm 1.

Example Consider an RFID system with 8 tags that uses skip lists with $k = 2$ and $\eta = 3$ for key assignment as shown in Figure 2. Tags are mapped to the t-th node in L_3. We illustrate how the key issuer assigns group keys and random numbers to a tag, for instance Tag 3. Starting from v_3, the key issuer traverses to the top level list. First, Tag 3 obtains sk_3 stored at $v_3.key[3]$, and the pointer moves to Level 2 via v_2. Because v_3 does not appear in L_2, the pointer goes to v_2 $(3 - 3 \ mod \ k = 2)$, and then moves to Level 2. Assume the key issuer randomly selects $r_2 = 3$ and the pointer shifts

Algorithm 1 Key Issue

```
/* Key Issuer does following */
Issuer locates all tags t to node v_i
/* For each tag i Key Issuer does following */
for for each tag t in the system do
    KeyIssue(i, v_i)
end for
/* The function to assign keys to Tag t */
KeyIssue(t, v_i)
/* v_i is the current node */
R_t = φ /* Initialize the random numbers list */
GK_t = φ /* Initialize the grop keys list */
/ * At the lowest level list L_η */
sk_t ← v_i.key[η]
v_i ← v_m where m = i − i mod k
for (j from η − 1 to 1) do
    /* Random shifting by r and add a group key */
    r ←(uniform)— [0, |L_j| − 1]
    Add r to R_t
    v_i ← shift to the left by r
    Add v.key[j] to GK_t
    /* Move to upper level */
    v_i ← v_m where m = i − i mod k^{η−j+1}
    j = j − 1
end for
```

to the left by 3. At the same time, 2 is added to R_3. The current pointer is now at v_4 in L_2. The issuer assigns $gk_{4,2}$ stored in $v_4.key[2]$ to Tag 3. This process continues until the issuer reaches L_0. Assume Tag 3 selects $r_1 = 1$ at Level 1. It obtains sk_3, $GK_3 = \{gk_{0,1}, gk_{4,2}\}$, and $R_3 = \{1, 3\}$.

2.5 Authentication

After issuing keys, the reader can securely communicate with tags. In RSLA authentication protocol, the reader first sends a query with nonce n_r, then a tag generates a reply message with nonce n_t, and then the reader decrypts the tag's reply.

Assume Tag t has one unique key sk_t, a set of group keys $GK_t = \{gk_1, gk_2, ..., gk_{\eta-1}\}$, and a set of random numbers $R_t = \{r_1, r_2, ..., r_{\eta-1}\}$. On receiving a query with nonce n_r from the reader, Tag t generates a reply message with nonce n_t. Let $\beta = \{\beta_1, \beta_2, ..., \beta_{\eta-1}\}$ be the reply message. Here, β_i $(i \leq 1 \leq \eta)$ consists of a hash value $\beta_i.hash$ and encrypted number $\beta_i.num$ at each level i. The hash value $\beta_i.hash$ is obtained by $H(gk_i||r_{i-1}||n_t||n_r)$ with the base

$r_0 = empty$. In other words, $\beta_1.hash = H(gk_1||n_t||n_r)$ because there is no rotation at L_0. The reason that we include the number at the previous level, i.e., r_{i-1} for $\beta_i.hash$, is to enforce dependency between the levels to keep high anonymity. The random number $\beta_i.num$ is encrypted by $E(gk_i, r_i)$. For the last element β_η, the hash value $\beta_\eta.hash$ is defined by $H(sk_t||r_{\eta-1}||n_t||n_r)$ where the unique key is used, and $\beta_\eta.num$ is empty. Finally, the tag sends n_t and β to the reader. Note that β contains η elements. One is computed by sk; the other $\eta-1$ are computed by gk. The pseudo code of the replying process is illustrated in Algorithm 2.

Algorithm 2 ReplyToReader(n_r)

```
/* Assume Tag t has sk_t, GK_t, and R_t */
/* where GK_t = {gk_1, gk_2, ..., gk_{η-1}} */
/* and R_t = {r_1, r_2, ..., r_{η-1}} */
Generate nonce n_t
for i from 1 to η − 1 do
    β_i.hash ← H(gk_i||r_{i-1}||n_t||n_r) /* r_0 = empty */
    β_i.num ← E(gk_i, r_i)
    Add β_i to β
end for
β_η.hash = H(sk_t||r_{η-1}||n_t||n_r)
reply n_t and β
```

Algorithm 3 Authentication(n_r, n_t, β)

```
/* β = {β_1, β_2, ..., β_η}*/
v_0 ← head /* the pointer to the current node */
for j from 1 to η do
    /* Scan v.key[j] for k nodes from v_i */
    for m from 1 to k do
        /* Note that the base r_0 = empty */
        if H(v_i.key[j]||r_{j-1}||n_r||n_t) = β_j.hash then
            if j == η then
                Identify Tag t by the unique key v_i.key[j]
            else
                r ← D(v_i.key[j], β_j.num)
                v_i ← shift to the right by r
                j ← j + 1
            end if
        end if
        m ← m + 1
    end for
    if The key is not found for L_j then
        return FAIL
    end if
end for
return t
```

On receiving Tag i's reply, the reader scans group keys associated to nodes from the top level list. At the beginning, the pointer is at node v_0 in L_0. In L_1, there are k nodes, and one of them has the group key $v_i.key[1]$ ($v_i \in L_1$) that matches the group key used for $\beta_1.hash$. After finding the corresponding key used for $\beta_1.hash$, the reader decrypts $\beta_1.num$ with the key. Then, we first move the pointer to L_1 form L_0, and shift the pointer to the right by $\beta_1.num$. If the pointer reaches the tail during shifting, it moves to the head of the same list. Note that the left shift was taken for key assignment by traveling from the lowest level, and on the contrary, the authentication process takes the right shift since the reader travels skip lists from the top. Assume v_i is the current node after shifting right by r_1. The list L_2 has k^2 nodes, but only k nodes v_j ($i \le j \le i+k$) need to be scanned. This is because one of the k nodes has the group key for β_2. This process continues until the reader reaches the bottom. Since the key at L_η is unique for a tag, the reader singulates the tag from β. The reader scans no more than k keys at each level $1 \le i \le \eta$, hence our skip lists imitate the search operation of a k-balanced tree. During this process, should the reader be unable to find a group key at any level, the tag's reply is invalid and the reader returns a $FAIL$ message. The pseudo code of the authentication process is provided in Algorithm 3.

3. CONCLUSION

Large-scale RFID systems always have tradeoffs between performance and security/privacy. The private authentication protocols proposed in the past are either slow or vulnerable to active attacks. In this paper, we propose RSLA which provides both high authentication efficiency and a strong privacy protection mechanism. RSLA relies on skip lists, a different data structure from the existing solutions. We believe the proposed skip lists-based approach is the most suitable authentication scheme for the next generation RFID systems.

4. REFERENCES

[1] G. Avoine, L. Buttyan, T. Holczer, and I. Vajda. Group-based Private Authentication. In *WoWMoM*, pages 1–6, 2007.

[2] T. Dimitriou. A Secure and Efficient RFID Protocol that cold make Big Brother (partially) Obsolete. In *PerCom*, pages 269–275, 2006.

[3] M. E. Hoque, F. Rahman, and S. I. Ahamed. AnonPri: An Efficient Anonymous Private Authentication Protocol. In *PerCom*, pages 102–110, 2011.

[4] A. Juels. Minimalist Cryptography for Low-Cost RFID Tags. In *SCN*, pages 149–164, 2004.

[5] W.-S. Ku, K. Sakai, and M.-T. Sun. The Optimal k-Covering Tag Deployment for RFID-Based Localization. *Special Issues of JNCA on RFID Technology, Systems, and Applications*, 34(3):914–924, 2011.

[6] L. Lu, J. Han, L. Hu, Y. Liu, and L. M. Ni. Dynamic Key-Updating: Privacy-Preserving Authentication for RFID Systems. In *PerCom*, pages 13–22, 2007.

[7] D. Molnar and D. Wagner. Privacy and Security in Library RFID Issues, Practices, and Architectures. In *CCS*, pages 210–219, 2004.

[8] W. Pugh. Skip Lists: a Probabilistic Alternative to Balanced Trees. *Comms. of the ACM*, 33(6):668–676, 1990.

[9] K. Sakai, W.-S. Ku, R. Zimmermann, and M.-T. Sun. Dynamic Bit Encoding for Privacy Protection against Correlation Attacks in RFID Backward Channel. *IEEE Trans. Computers*, 62(1):112–123, 2013.

[10] K. Sakai, M.-T. Sun, W.-S. Ku, T. H. Lai, and A. V. Vasilakos. Randomized Skip Lists-Based Private Authentication for Large-Scale RFID Systems. *OSU-CISRC-5/13-TR12*, 2013. ftp://ftp.cse.ohio-state.edu/pub/tech-report/2013/TR12.pdf.

[11] S. Wagner, M. Handte, M. Zuniga, and P. J. Marron. On Optimal Tag Placement for Indoor Localization. In *PerCom*, pages 162–170, 2012.

[12] J. Yu, W.-S. Ku, M.-T. Sun, and H. Lu. An RFID and particle filter-based indoor spatial query evaluation system. In *EDBT*, pages 263–274, 2013.